VERITAS™

VERITAS Volume Replicator 4.1

Planning and Tuning Guide

HP-UX

N13012G

June 2005

VERITAS Software Corporation
350 Ellis Street
Mountain View, CA 94043
USA
Phone 650–527–8000 Fax 650–527–2908
www.veritas.com

Contents

Planning and Configuring Replication 1

To set up an efficient VERITAS™ Volume Replicator (VVR) configuration, it is necessary to understand how the various VVR components interact with each other. This chapter explains the interactions and presents the decisions you must make when setting up a VVR configuration. The chapter "Tuning Replication Performance" on page 25 explains performance considerations. This document assumes that you understand the concepts of VVR. For more information, read the description of concepts in the *VERITAS Volume Replicator Administrator's Guide*.

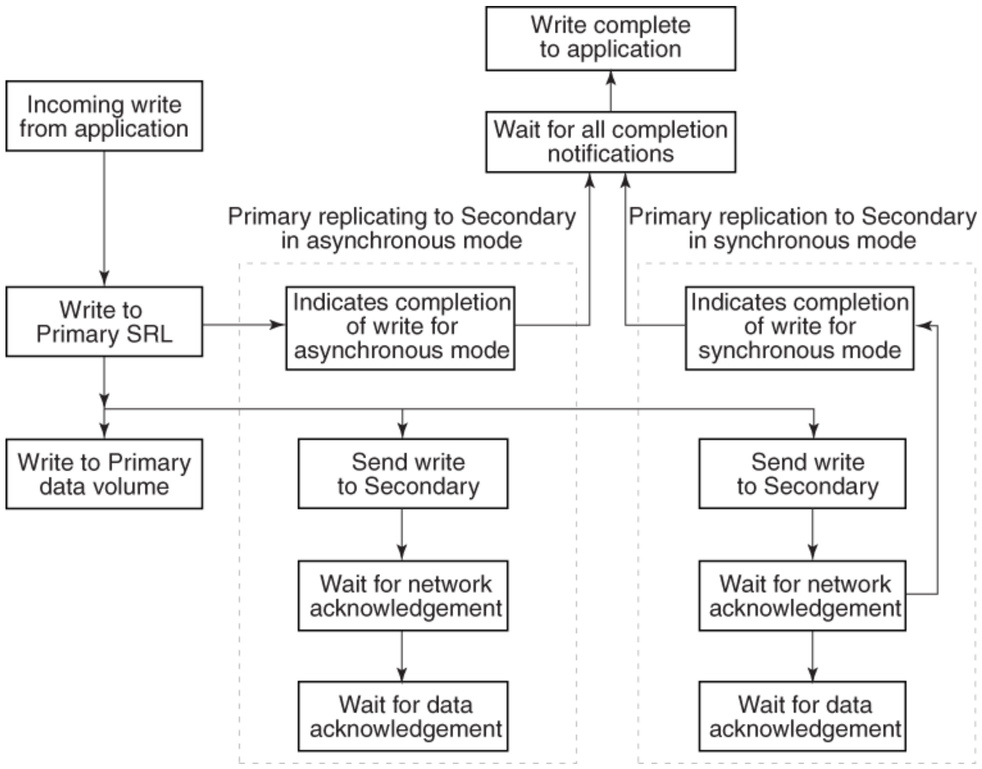
In an ideal configuration, data is replicated at the speed at which it is generated by the application. As a result, all Secondary hosts remain up to date. A write to a data volume in the Primary flows through various components and across the network until it reaches the Secondary data volume. For the data on the Secondary to be up to date, each component in the configuration must be able to keep up with the incoming writes. The goal when configuring replication is that VVR be able to handle temporary bottlenecks, such as occasional surges of writes, or occasional network problems.

If one of the components cannot keep up with the write rate over the long term, the application could slow down because of increased write latency, the Secondary could fall behind, or the SRL might overflow. If a component on the path that completes the write on the Primary cannot keep up, latency might be added to each write, which leads to low application performance. If other components, which are not on this path, cannot keep up with the write rate, it is likely that the writes on the Primary proceed at their normal pace but accumulate in the SRL. As a result, the Secondary falls behind and the SRL eventually overflows. Therefore, it is important to examine each component to ensure that it can support the expected application write rate.

In this document, the term *application* refers to the program that writes directly to the data volume. If a database is using a file system mounted on a data volume, the file system is the application; if the database writes directly to a data volume, then it is considered the application. For information about the VVR concepts, refer to the *VERITAS Volume Replicator Administrator's Guide*.

Data Flow in VVR

This section explains how data flows in VVR and how VVR uses the kernel buffers for replication. The figure "Data flow with multiple Secondary hosts" shows the flow of data for a VVR configuration containing two Secondary hosts with the Primary replicating to one host in asynchronous mode and the other host in synchronous mode.



Write complete to application
Incoming write from application
Wait for all completion notifications
Primary replicating to Secondary in asynchronous mode
Primary replication to Secondary in synchronous mode
Write to Primary SRL
Indicates completion of write for asynchronous mode
Indicates completion of write for synchronous mode
Write to Primary data volume
Send write to Secondary
Send write to Secondary
Wait for network acknowledgement
Wait for network acknowledgement
Wait for data acknowledgement
Wait for data acknowledgement

Data flow with multiple Secondary hosts

When a write is performed on a data volume associated with a Replicated Volume Group (RVG), VVR copies the data into a kernel buffer on the Primary. VVR then writes a header and the data to the SRL; the header describes the write.

From the kernel buffer, VVR sends the write to all Secondary hosts and writes it to the Primary data volume. Writing the data to the Primary data volume is performed asynchronously to avoid adding the penalty of a second full disk write to the overall write latency. Until the data volume write to the Primary is complete, the kernel buffer cannot be freed.

For all Secondary hosts replicating in synchronous mode, VVR first sends the write to the Primary SRL. VVR then sends the write to the Secondary hosts and waits for a network acknowledgement that the write was received. When all Secondary hosts replicating in synchronous mode have acknowledged the write, VVR notifies the application that the write is complete. The Secondary sends the network acknowledgement as soon as the write is received in the VVR kernel memory on the Secondary. The application does not need to wait for the full disk write, which improves performance. The data is subsequently written to the Secondary data volumes. When the write is completed on the Secondary data volumes, VVR sends a data acknowledgement back to the Primary.

For all Secondary hosts replicating in asynchronous mode, VVR notifies the application that the write is complete after it is written to the Primary SRL. Therefore, the write latency consists of the time to write to the SRL only. VVR then sends the write to the Secondary hosts. The Secondary sends a network acknowledgement to the Primary as soon as the write is received in the VVR kernel memory on the Secondary. When the write is completed on the Secondary data volumes, VVR sends a data acknowledgement back to the Primary.

The application considers the write complete after receiving notification from VVR that the data is written to the Primary SRL, and, for any Secondary hosts replicating in synchronous mode, that the write has been received in the kernel buffer. However, VVR continues to track the write until the data acknowledgement is received from all the Secondary hosts. If the Secondary crashes before writing to the data volumes on the Secondary or if the Primary crashes before it receives the data acknowledgement, the write can be replayed from the SRL.

Data Flow When Reading Back from the SRL

A Secondary in asynchronous mode might be out of date for various reasons, such as network outages or a surge of writes which exceed available network bandwidth. As the Secondary falls behind, the data to be sent to the Secondary starts accumulating in the write-buffer space on the Primary. If the Secondaries in asynchronous mode cannot keep up with the application write rate, VVR might need to free the Primary kernel buffer, so that incoming write requests are not delayed.

Secondary hosts that fall behind in this manner are serviced by reading back the writes from the Primary SRL. In this case, the writes are sent from the Read Back Buffer, rather than from the Primary buffer as described earlier. The read back process continues until the Secondary catches up with the Primary; at this point, the process of sending writes to the Secondary reverts back to sending from the kernel buffer, instead of sending by reading back from the SRL.

Before You Begin Configuring

Before you begin configuring VVR, you must understand the characteristics of the application writes that are to be replicated. You must also understand the needs of the business for which VVR is being deployed.

Understanding Business Needs

To satisfy the needs of your business, you must consider the following:

- The amount of data that can be lost if a disaster occurs and yet continue the business successfully
- The amount of time acceptable to recover the data after the disaster and continue the business successfully

In a traditional tape backup scheme, the amount of data lost in a disaster can be large, depending on the frequency of backup and tape vaulting. Also, the recovery time from a tape backup can be significant. In a VVR environment, recovery time is negligible and the amount of data lost depends on the following factors:

- Mode of replication
- Network bandwidth
- Network latency between the Primary and the Secondary
- Ability of the Secondary data volumes to keep up with the write rate

If the data on the Secondary must be as up to date as possible, we recommend that you use synchronous mode and provide the same bandwidth as the peak rate at which the application writes on the Primary. However, if the Secondary can be allowed to lag behind, we recommend that you use asynchronous mode and provide the same bandwidth as the average rate at which the application writes on the Primary. These decisions are determined by your business needs.

Understanding Application Characteristics

Before you configure an RDS, you must know the data throughput that must be supported, that is, the rate at which the application can be expected to write data. Only write operations are of concern; read operations do not affect replication. To perform the analyses described in later sections, a profile of application write rate is required. For an application with relatively constant write rate, the profile could take the form of certain values, such as:

- Average application write rate
- Peak application write rate
- Period of peak application write rate

For a more volatile application, a table of measured usages over specified intervals may be needed. Because matching application write rate to disk capacity is not an issue unique to replication, it is not discussed here. It is assumed that an application is already running, and that VERITAS Volume Manager (VxVM) has been used to configure data volumes to support the write rate needs of the application. In this case, the application write rate characteristics may already have been measured.

If the application characteristics are not known, they can be measured by running the application and using a tool to measure data written to all the volumes to be replicated. If the application is writing to a file system rather than a raw data volume, be careful to include in the measurement all the metadata written by the file system as well. This can add a substantial amount to the total amount of replicated data. For example, if a database is using a file system mounted on a replicated volume, a tool such as `vxstat` (see `vxstat`(1M)) correctly measures the total data written to the volume, while a tool that monitors the database and measures its requests fails to include those made by the underlying file system.

It is also important to consider both peak and average write rates of the application. These numbers can be used to determine the type of network connection needed. For Secondary hosts replicating in synchronous mode, the network must support the peak application write rate. For Secondary hosts replicating in asynchronous mode that are not required to keep pace with the Primary, the network only needs to support the average application write rate.

Finally, once the measurements are made, the numbers calculated as the peak and average write rates should be close to the largest obtained over the measurement period, not the averages or medians. For example, assume that measurements are made over a 30-day period, yielding 30 daily peaks and 30 daily averages, and then the average of each of these is chosen as the application peak and average respectively. If the network is sized based on these values, then for half the time there will be insufficient network capacity to keep up with the application. Instead, the numbers chosen should be close to the highest obtained over the period, unless there is reason to doubt that they are valid or typical.

Choosing the Mode of Replication

The decision to use synchronous or asynchronous mode must be made with a complete understanding of the effects of this choice on application and replication performance. The relative merits of using synchronous or asynchronous mode become apparent when you understand the underlying process of replication.

Synchronous Mode Considerations

Synchronous mode has the advantage that all writes are guaranteed to reach the Secondary before completing. For some businesses, this may simply be a requirement that cannot be circumvented – in this case, performance is not a factor in the decision. For applications where the choice is not so clear, however, this section discusses some of the performance implications of choosing synchronous operations.

As illustrated in the figure "Data flow with multiple Secondary hosts" on page 2, all write requests first result in a write to the SRL. It is only after this write completes that data is sent to the Secondary. Because synchronous mode requires that the data reach the Secondary and be acknowledged before the write completes, this makes the latency for a write equal to:

```
SRL latency + Network round trip latency
```

Thus, synchronous mode can significantly decrease application performance by adding the network round trip to the latency of each write request.

If you choose synchronous mode, you must consider what VVR should do if there is a network interruption. In synchronous mode, the `synchronous` attribute enables you to specify what action is taken when the Secondary is unreachable. The `synchronous` attribute can be set to `override` or `fail`. When the `synchronous` attribute is set to `override`, synchronous mode converts to asynchronous during a temporary outage. In this case, after the outage passes and the Secondary catches up, replication reverts to synchronous.

When the `synchronous` attribute is set to `fail`, the application receives a failure for writes issued while the Secondary is unreachable. The application is likely to fail or become unavailable, and hence this setting must be chosen only if such a failure is preferable to the Secondary being out of date.

We recommend setting the `synchronous` attribute to `override`, as this behavior is suitable for most applications. Setting the `synchronous` attribute to `fail` is suitable only for a special class of applications that cannot have even a single write difference between the Primary and Secondary data volumes. In other words, this mode of operation must be used only if you want an application write to fail if the write cannot be replicated immediately. It is imperative that the network connection between hosts using this option must be highly reliable to avert unnecessary application downtime as network outage could cause an application outage.

Additional Considerations When the synchronous Attribute Is Set to fail

When the `synchronous` attribute is set to `fail`, VVR ensures that writes do not succeed if they do not reach the Secondary. If the RLINK is disconnected, the writes fail and are not written either to the SRL or the data volumes. However, if the RLINK was connected but disconnects during the process of sending the writes to the Secondary, it is possible that the writes are written into the SRL and applied to the data volumes even though the application correctly receives failure for these writes. This happens because the data volume writes are asynchronous regardless of the mode of replication. For more information about the sequence of operations, see “Data Flow in VVR” on page 2.

The state of the running application on the Primary at this time is no different from that of the application brought up on the Secondary after changing its role to Primary. However, the actual contents of the Primary data volumes and the Secondary data volumes differ, and the Primary data volumes are ahead by these last writes.

Note that as soon as the synchronous RLINK connects, these writes will reach the Secondary, and then the data volumes on the Primary and the Secondary have the same contents. Also, note that at no time is the data consistency being compromised.

If the application is stopped or crashes at this point and is restarted, it recovers using the updated contents of the data volumes.The behavior of the application on the Primary could be different from the behavior of the application when it is brought up on the Secondary after changing its role of the Secondary to Primary, while the RLINK was still disconnected.

In the case of a database application, these writes might be the ones that commit a transaction. If the application tries to recover using the data volumes on the Primary, it will roll forward the transaction because the commit of the transaction is already on the data volume. However, if the application recovers using the data volumes on the Secondary after changing its role to Primary, it will roll back the transaction.

This case is no different from that of an application directly writing to a disk that fails just as it completes part of a write. Part of the write physically reaches the disk but the application receives a failure for the entire write. If the part of the write that reached the disk is the part that is useful to the application to determine whether to roll back or roll forward a transaction, then the transaction would succeed on recovery even though the transaction was failed earlier.

It could also happen that a write was started by the application and the RLINK disconnected and now before the next write is started, the RLINK reconnects. In this case, the application receives a failure for the first write but the second write succeeds.

Different applications, such as file systems and databases, deal with these intermittent failures in different ways. The VERITAS File System handles the failure without disabling the file or the file system.

Asynchronous Mode Considerations

Asynchronous mode of replication avoids adding the network latency to each write by sending the data to the Secondary after the write is completed to the application. The obvious disadvantage of this is that there is no immediate guarantee that a write that appears complete to the application has actually been replicated. A more subtle effect of asynchronous mode is that while application throughput remains mostly unaffected, overall replication performance may decrease.

In asynchronous mode, the Primary kernel memory buffer fills up if the network bandwidth or the Secondary cannot keep up with the incoming write rate. For VVR to provide memory for incoming writes and continue their processing, it must free the memory held by writes that have been written to the Primary data volume but not yet sent to the Secondary. When VVR is ready to send the unsent writes that were freed, the writes must first be read back from the SRL. Hence, in synchronous mode the data is always available in memory, while in asynchronous mode VVR might have to frequently read back the data from the SRL. Consequently, replication performance might suffer because of the delay of the additional read operation.

The need to read back from the SRL has a negative impact on the SRL performance. In synchronous mode, VVR does not need to read back data from the SRL but only performs sequential writes, which results in better performance. In asynchronous mode, the writes may be interspersed with occasional reads from the SRL and hence performance may deteriorate. VVR does not need to read back from the SRL if the network bandwidth and the Secondary always keep up with the incoming write rate, or if the Secondary only falls behind for short periods during which the accumulated writes are small enough to fit in the VVR kernel buffer.

For information on how to tune the size of kernel buffers for VVR and VxVM, see “VVR Buffer Space” on page 27. If VVR reads back from the SRL frequently, striping the SRL over several disks using mid-sized stripes (for example, 10 times the average write size), could improve performance. To determine whether VVR is reading back from the SRL, use the `vxstat` command. In the output, note the number of read operations on the SRL.

Asynchronous Replication Versus Synchronous Replication

The decision to use synchronous or asynchronous replication depends on the requirements of your business and the capabilities of your network. The main considerations are summarized in the following table.

Note If you have multiple Secondaries, you can have some replicating in asynchronous mode and some in synchronous mode. For more information, see "How Data Flows in an RDS Containing Multiple Secondary Hosts" in the *VERITAS Volume Replicator Administrator's Guide*.

Considerations	
Synchronous Mode	**Asynchronous Mode**
Need for Secondary to be up-to-date	
Ensures that the Secondary is always current. If the synchronous attribute is set to override, the Secondary is current, except in the case of a network outage.	Ensures that the Secondary reflects the state of the Primary at some point in time. However, the Secondary may not be current. The Primary may have committed transactions that have not been written to the Secondary.
Requirements for managing latency of data	
Works best for low volume of writes. Does not require latency protection (because the Secondary is always current).	Could result in data latency on the Secondary. You need to consider whether or not it is acceptable to lose committed transactions if a disaster strikes the Primary, and if so, how many. VVR enables you to manage latency protection, by specifying how many outstanding writes are acceptable, and what action to take if that limit is exceeded.
Characteristics of your network: bandwidth, latency, reliability	
Works best in high bandwidth/low latency situations. If the network cannot keep up, the application may be impacted. Network capacity should meet or exceed the write rate of the application at all times.	Handles bursts or congestion on the network by using the SRL. This minimizes impact on application performance from network bandwidth fluctuations. The average network bandwidth must be adequate for the average write rate of the application. Asynchronous replication does not compensate for a slow network.

Considerations	
Synchronous Mode	**Asynchronous Mode**
Requirements for application performance, such as response time.	
Has potential for greater impact on application performance because the I/O does not complete until the network acknowledgement is received from the Secondary.	Minimizes impact on application performance because the I/O completes without waiting for the network acknowledgment from the Secondary.

Choosing Latency and SRL Protection

The replication parameters `latencyprot` and `srlprot` provide a compromise between synchronous and asynchronous characteristics. These parameters allow the Secondary to fall behind, but limit the extent to which it does so.

When `latencyprot` is enabled, the Secondary is only allowed to fall behind by a predefined number of requests, a latency high mark. After this user-defined latency high mark is reached, throttling is triggered. This forces all incoming requests to be delayed until the Secondary catches up to within another predefined number of requests, the latency low mark. Thus, the average write latency seen by the application increases. A large difference between the latency high mark and latency low mark causes occasional long delays in write requests, which may appear to be application hangs, as the SRL drains down to the latency low mark. Most other write requests remain unaffected. A smaller range spreads the delays more evenly over writes, resulting in smaller but more frequent delays. For most cases, a smaller difference is probably preferable.

The replication parameter `srlprot` can be used to prevent the SRL from overflowing and has an effect similar to `latencyprot`. However, the `srlprot` attribute is set to `autodcm` by default, which allows the SRL to overflow and convert to `dcm_logging` mode. As a result, there is no effect on write performance, but the Secondary is allowed to fall behind and is inconsistent while it resynchronizes. For more information, refer to the *VERITAS Volume Replicator Administrator's Guide*.

Planning the Network

This section describes the available network protocols for replication in VVR. It also explains how bandwidth requirement depends on the mode of replication—synchronous or asynchronous.

Choosing the Network Bandwidth

Bandwidth of the Available Network Connection

The type of connection determines the maximum bandwidth available between the two locations, for example, a T3 line provides 45 megabits/second. However, the important factor to consider is whether the available connection is to be used by any other applications or is exclusively reserved for replicating to a single Secondary. If other applications are using the same line, it is important to be aware of the bandwidth requirements of these applications and subtract them from the total network bandwidth. If any applications sharing the line have variations in their usage pattern, it is also necessary to consider whether their times of peak usage are likely to coincide with peak network usage by VVR. Additionally, overhead added by VVR and the various underlying network protocols reduces effective bandwidth by a small amount, typically 3% to 5%.

How Network Performance Depends on Mode of Replication

All replicated write requests must eventually travel over the network to one or more Secondary nodes. Whether or not this trip is on the critical path depends on the mode of replication.

Because replicating in synchronous mode requires that data reach the Secondary node before the write can complete, the network is always part of the critical path for synchronous mode. This means that for any period during which application write rate exceeds network capacity, write latency increases.

Conversely, replicating in asynchronous mode does not impose this requirement, so write requests are not delayed if network capacity is insufficient. Instead, excess requests accumulate on the SRL, as long as the SRL is large enough to hold them. If there is a persistent shortfall in network capacity, the SRL eventually overflows. However, this setup does allow the SRL to be used as a buffer to handle temporary shortfalls in network capacity, such as periods of peak usage, provided that these periods are followed by periods during which the Secondary can catch up as the SRL drains. If a configuration is planned with this functionality in mind, you must be aware that Secondary sites may be frequently out of date.

Several parameters can change the asynchronous mode behavior described above by placing the network round-trip on the critical path in certain situations. The `latencyprot` and `srlprot` features, when enabled, can both have this effect. These features are discussed fully in "Choosing Latency and SRL Protection" on page 11.

To avoid problems caused by insufficient network bandwidth, apply the following principles:

- If synchronous mode is used, the network bandwidth must at least match the application write rate during its peak usage period; otherwise, the application is throttled. However, this leaves excess capacity during non-peak periods, which is useful to allow synchronization of new volumes using checkpoints as described in "Peak Usage Constraint" on page 18.
- If only asynchronous mode is used, and you have the option of allowing the Secondary to fall behind during peak usage, then the network bandwidth only needs to match the overall average application write rate. This might require the application to be shut down during synchronization procedures, because there is no excess network capacity to handle the extra traffic generated by the synchronization.
- If asynchronous mode is used with `latencyprot` enabled to avoid falling too far behind, the requirements depend on how far the Secondary is allowed to fall behind. If the latency high mark is small, replication will be similar to synchronous mode and therefore must have a network bandwidth sufficient to match the application write rate during its peak usage period. If the latency high mark is large, the Secondary can fall behind by several hours. Thus, the bandwidth only has to match the average application write rate.

Choosing the Network Protocol

VVR exchanges two types of messages between the Primary and the Secondary: heartbeat messages and data messages. The heartbeat messages are transmitted using the UDP transport protocol. VVR can use either the TCP transport protocol or the UDP transport protocol to exchange data messages.

The choice of protocol to use for the data messages is based on the network characteristics. TCP has been found to perform better than UDP on networks that lose packets. However, you must experiment with both protocols to determine the one that performs better in your network environment.

Note You *must* specify the same protocol for the Primary and Secondary; otherwise, the nodes cannot communicate and the RLINKs do not connect. This also applies to all nodes in a cluster environment.

VVR uses the UDP transport protocol by default.

For information on how to set the network protocol, refer to the *VERITAS Volume Replicator Administrator's Guide.*

Choosing the Network Ports Used by VVR

VVR uses the UDP and TCP transport protocols to communicate between the Primary and Secondary. This section lists the default ports used by VVR.

By default, VVR uses the following ports when replicating data using UDP.

Port Numbers	**Description**
UDP 4145	IANA approved port for heartbeat communication between the Primary and Secondary.
TCP 8199	IANA approved port for communication between the `vradmind` daemons on the Primary and the Secondary.
TCP 8989	Communication between the `in.vxrsyncd` daemons, which are used for differences-based synchronization.
UDP Anonymous ports (OS dependent)	Ports used for each Primary-Secondary connection for data replication between the Primary and the Secondary. One data port is required on each host.

By default, VVR uses the following ports when replicating data using TCP.

Port Numbers	Description
UDP 4145	IANA approved port for heartbeat communication between the Primary and Secondary.
TCP 4145	IANA approved port for TCP Listener port.
TCP 8199	IANA approved port for communication between the `vradmind` daemons on the Primary and the Secondary.
TCP 8989	Communication between the `in.vxrsyncd` daemons, which are used for differences-based synchronization.
TCP Anonymous ports	Ports used for each Primary-Secondary connection for data replication between the Primary and the Secondary. One data port is required on each host.

The `vrport` command enables you to view and change the port numbers used by VVR. For instructions, see the *VERITAS Volume Replicator Administrator's Guide*.

Configuring VVR in a Firewall Environment

This section explains how to configure VVR to work in a firewall environment. For information about the default port numbers used by VVR, see section, "Choosing the Network Ports Used by VVR" on page 14. For information about replicating over a Network Address Translation (NAT) based firewall, see section "VVR and Network Address Translation Firewall" on page 35.

To configure VVR in a firewall environment when using TCP:

Enable in the firewall the following ports: the port used for heartbeats, the port used by the `vradmind` daemon, and the port used by the `in.vxrsyncd` daemon. Use the `vrport` command to display information about the ports and to change the ports being used by VVR.

To configure VVR in a firewall environment when using UDP:

1. In the firewall, enable the following ports:

   - the port used for heartbeats
   - the port used by the `vradmind` daemon and
   - the port used by the `in.vxrsyncd` daemon.

   Use the `vrport` command to display information about the ports and to change the ports being used by VVR.

2. Set a restricted number of ports to replicate data between the Primary and the Secondary. The operating system assigns anonymous port numbers by default. Most operating systems assign anonymous port numbers between 32768 and 65535. For each Primary-Secondary connection, one data port is required. Use the `vrport` command to specify a list of ports or range of ports to use for VVR.

3. In the firewall, enable the ports that have been set in step 2.

Choosing the Packet Size

If you have selected the UDP transport protocol for replication, the UDP packet size used by VVR to communicate between hosts could be an important factor in the replication performance. By default, VVR uses a UDP packet size of 8400 bytes. In certain network environments, such as those that do not support fragmented IP packets, it may be necessary to change the packet size. For instructions on how to change the packet size, see the *VERITAS Volume Replicator Administrator's Guide*.

Choosing the Network Maximum Transmission Unit

The UDP packets or TCP packets transmitted by VVR that are of size greater than the network Maximum Transmission Unit (MTU) are broken up into IP packets of MTU size by the IP module of the operating system. Certain network environments, such as Virtual Private Networks (VPNs), may add information to each IP packet. This makes the size of the IP packet greater than the network MTU, thus resulting in additional fragmentation. This results in twice the number of packets on the network. To avoid the second fragmentation, identify the number of bytes added to each packet by the VPN controller and reduce the network MTU by that amount.

Sizing the SRL

The size of the SRL is critical to the performance of replication. This section describes some of the considerations in determining the size of the SRL. Refer also to the *VERITAS Volume Replicator Advisor User's Guide* for information about using the Volume Replicator Advisor (`VRAdvisor`) tool to help determine the appropriate SRL size.

When the SRL overflows for a particular Secondary, the RLINK corresponding to that Secondary is marked STALE and becomes out of date until a complete resynchronization with the Primary is performed. Because resynchronization is a time-consuming process and during this time the data on the Secondary cannot be used, it is important to avoid SRL overflows. The SRL size needs to be large enough to satisfy four constraints:

- It must not overflow for asynchronous RLINKs during periods of peak usage when replication over the RLINK may fall far behind the application.
- It must not overflow while a Secondary RVG is being synchronized.
- It must not overflow while a Secondary RVG is being restored.
- It must not overflow during extended outages (network or Secondary node).

Note The size of the SRL must be at least 110 MB. If the size that you have specified for the SRL is less than 110 MB, VVR displays an error message which prompts you to specify a value that is equal to or greater then 110 MB.

To determine the size of the SRL, you must determine the size required to satisfy each of these constraints individually. Then, choose a value at least equal to the maximum so that all constraints are satisfied. The information needed to perform this analysis, presented below, includes:

- The maximum expected downtime for Secondary nodes
- The maximum expected downtime for the network connection
- The method for synchronizing Secondary data volumes with data from Primary data volumes. If the application is shut down to perform the synchronization, the SRL is not used and the method is not important. Otherwise, this information could include: the time required to copy the data over a network, or the time required to copy it to a tape or disk, to send the copy to the Secondary site, and to load the data onto the Secondary data volumes.

Note If the Automatic Synchronization option is used to synchronize the Secondary, the previous paragraph is not a concern.

If you are going to perform Secondary backup to avoid complete resynchronization in case of Secondary data volume failure, the information needed also includes:

- The frequency of Secondary backups
- The maximum expected delay to detect and repair a failed Secondary data volume
- The expected time to reload backups onto the repaired Secondary data volume

Peak Usage Constraint

For some configurations, it might be common for replication to fall behind the application during some periods and catch up during others. For example, an RLINK might fall behind during business hours and catch up overnight if its peak bandwidth requirements exceed the network bandwidth. Of course, for synchronous RLINKs, this does not apply, as a shortfall in network capacity would cause each application write to be delayed, so the application would run more slowly, but would not get ahead of replication.

For asynchronous RLINKs, the only limit to how far replication can fall behind is the size of the SRL. If it is known that the peak write rate requirements of the application exceed the available network bandwidth, then it becomes important to consider this factor when sizing the SRL.

Assuming that data is available providing the typical application write rate over a series of intervals of equal length, it is simple to calculate the SRL size needed to support this usage pattern:

1. Calculate the network capacity over the given interval (BW_N).
2. For each interval *n*, calculate SRL log volume growth (LG_n), as the excess of application write rate (BW_{AP}) over network bandwidth ($LG_n = BW_{AP(n)} - BW_N$).
3. For each interval, accumulate all the SRL growth values to find the cumulative SRL log size (LS):

$$LS_n = \sum_{i=1...n} LG_i$$

The largest value obtained for any LS_n is the value that should be used for SRL size as determined by the peak usage constraint. See the table "Example Calculation of SRL Size Required to Support Peak Usage Period" on page 19, which shows an example of this calculation. The third column, Application, contains the maximum likely application write rate per hour obtained by measuring the application as discussed in "Understanding Application Characteristics" on page 5. The fourth column, Network, shows the network bandwidth. The fifth column, SRL Growth, shows the difference between application write rate and network bandwidth obtained for each interval. The sixth column, Cumulative SRL Size, shows the cumulative difference every hour. The largest value in column 6 is 37 gigabytes. The SRL should be at least this large for this application.

Note that several factors can reduce the maximum size to which the SRL can grow during the peak usage period. Among these are:

- The `latencyprot` characteristic can be enabled to restrict the amount by which the RLINK can fall behind, slowing down the write rate.
- The network bandwidth can be increased to handle the full application write rate. In this example, the bandwidth should be 15 gigabytes/hour—the maximum value in column three.

Example Calculation of SRL Size Required to Support Peak Usage Period

Hour Starting	**Hour Ending**	**Application (GB/hour)**	**Network (GB/hour)**	**SRL Growth (GB)**	**Cumulative SRL Size (GB)**
7am	8 a.m.	6	5	1	1
8	9	10	5	5	6
9	10	15	5	10	16
10	11	15	5	10	26
11	12 p.m.	10	5	5	31
12 p.m.	1	2	5	-3	28
1	2	6	5	1	29
2	3	8	5	3	32
3	4	8	5	3	35
4	5	7	5	2	37
5	6	3	5	-2	35

Synchronization Period Constraint

When a new Secondary is added to an RDS, its data volumes must be synchronized with those of the Primary unless the Primary and the Secondary data volumes have been zero initialized and the application has not yet been started. You also need to synchronize the Secondary after a Secondary data volume failure, in case of SRL overflow, or after replication is stopped.

This section applies if you choose *not* to use the automatic synchronization method to synchronize the Secondary. Also, this constraint does not apply if you choose to use a method other than automatic synchronization and if the application on the Primary can be shut down while the data is copied to the Secondary. However, in most cases, it might be necessary to synchronize the Secondary data volumes with the Primary data volumes while the application is still running on the Primary. This is performed using one of the methods described in the *VERITAS Volume Replicator Administrator's Guide*.

During the synchronization period, the application is running and data is accumulating in the SRL. If the SRL overflows during the process of synchronization, the synchronization process must be restarted. Thus, to ensure that the SRL does not overflow during this period, it is necessary that the SRL be sized to hold as much data as the application writes during the synchronization period. After starting replication, this data is replicated and the Secondary eventually catches up with the Primary.

Depending on your needs, it may or may not be possible to schedule the synchronization during periods of low application write activity. If it is possible to complete the synchronization process during a period of low application write activity, then you must ensure that the SRL is sized such that it can hold all the incoming writes during this period. Otherwise, the SRL may overflow. For more information on how to arrive at an optimum SRL size, refer to the *VERITAS Volume Replicator Adviser User's Guide*. If however there is an increase in the application write activity then you may need to resize the SRL even when the synchronization is in progress. For more information on resizing the SRL, see section "Resizing the SRL" in the *VERITAS Volume Replicator Administrator's Guide*. If it is not possible to complete the synchronization process during periods of low application write activity, then size the SRL such that it uses either the average value, or to be safer, the peak value. For more information, see section "Understanding Application Characteristics" on page 5.

Secondary Backup Constraint

VVR provides a mechanism to perform periodic backups of the Secondary data volumes. In case of a problem that would otherwise require a complete resynchronization using one of the methods described in"Synchronization Period Constraint" on page 20, a Secondary backup, if available, can be used to bring the Secondary online much more quickly.

A Secondary backup is made by defining a Secondary checkpoint and then making a raw copy of all the Secondary data volumes. Should a failure occur, the Secondary data volumes are restored from this local copy, and then replication proceeds from the checkpoint, thus replaying all the data from the checkpoint to the present.

The constraint introduced by this process is that the Primary SRL must be large enough to hold all the data logged in the Primary SRL after the creation of the checkpoint corresponding to the most recent backup. This depends largely on three factors:

- The application write rate.
- The frequency of Secondary backups.

Thus, given an application write rate and frequency of Secondary backups, it is possible to come up with a minimal SRL size. Realistically, an extra margin should be added to an estimate arrived at using these figures to cover other possible delays, including:

- Maximum delay before a data volume failure is detected by a system administrator.
- Maximum delay to repair or replace the failed drive.
- Delay to reload disk with data from the backup tape.

To arrive at an estimate of the SRL size needed to support this constraint, first determine the total time period the SRL needs to support by adding the period planned between Secondary backups to the time expected for the three factors mentioned above. Then, use the application write rate data to determine, for the worst case, the amount of data the application could generate over this time period.

Note Even if only one volume failed, all volumes must be restored.

Secondary Downtime Constraint

When the network connection to a Secondary node, or the Secondary node itself, goes down, the RLINK on the Primary node detects the broken connection and responds. If the RLINK has its `synchronous` attribute set to `fail`, the response is to fail all subsequent write requests until the connection is restored. In this case, the SRL does not grow, so the downtime constraint is irrelevant. For all other types of RLINKs, incoming write requests accumulate in the SRL until the connection is restored. Thus, the SRL must be large enough to hold the maximum output that the application could be expected to generate over the maximum possible downtime.

Maximum downtimes may be difficult to estimate. In some cases, the vendor may guarantee that failed hardware or network connections will be repaired within some period. Of course, if the repair is not completed within the guaranteed period, the SRL overflows despite any guarantee, so it is a good idea to add a safety margin to any such estimate.

To arrive at an estimate of the SRL size needed to support this constraint, first obtain estimates for the maximum downtimes which the Secondary node and network connections could reasonably be expected to incur. Then, use the application write rate data to determine, for the worst case, the amount of data the application could generate over this time period. With the introduction of the `autodcm` mode of SRL overflow protection, sizing the SRL for downtime is not essential to prevent SRL overflow because the changed blocks are no longer stored in the SRL. However, note that the Secondary is inconsistent during the replay of the DCM, and hence it is still important for the SRL to be large enough to cover most eventualities.

Additional Factors

Once estimates of required SRL size have been obtained under each of the constraints described above, several additional factors must be considered.

For the synchronization period, downtime and Secondary backup constraints, it is not unlikely that any of these situations could be immediately followed by a period of peak usage. In this case, the Secondary could continue to fall further behind rather than catching up during the peak usage period. As a result, it might be necessary to add the size obtained from the peak usage constraint to the maximum size obtained using the other constraints. Note that this applies even for synchronous RLINKs, which are not normally affected by the peak usage constraint, because after a disconnect, they act as asynchronous RLINKs until caught up.

Of course, it is also possible that other situations could occur requiring additions to constraints. For example, a synchronization period could be immediately followed by a long network failure, or a network failure could be followed by a Secondary node failure. Whether and to what degree to plan for unlikely occurrences requires weighing the cost of additional storage against the cost of additional downtime caused by SRL overflow.

Once an estimate has been computed, one more adjustment must be made to account for the fact that all data written to the SRL also includes some header information. This adjustment must take into account the typical size of write requests. Each request uses at least one additional disk block (1024) for header information, so the adjustments are as follows:

If Average Write Size is:	Add This Percentage to SRL Size:
1K	100%
2K	50%
4K	25%
8K	13%

If Average Write Size is:	Add This Percentage to SRL Size:
10K	10%
16K	6%
32K or more	3%

Example

This section shows how to calculate the SRL size for a VVR configuration. First, collect the relevant parameters for the site. For this site, they are as follows:

Application peak write rate	1 gigabyte/hour
Duration of peak	8 am - 8 pm
Application off-peak write rate	250 megabytes/hour
Average write size	2 kilobytes
Number of Secondary sites	1
Type of RLINK	synchronous=override
Synchronization Period:	
application shutdown	no
copy data to tape	3 hours
send tapes to Secondary site	4 hours
load data	3 hours
Total	10 hours
Maximum downtime for Secondary node	4 hours
Maximum downtime for network	24 hours
Secondary backup	not used

Because synchronous RLINKs are to be used, the network bandwidth must be sized to handle the peak application write rate to prevent the write latency from growing. Thus, the peak usage constraint is not an issue, and the largest constraint is that the network could be out for 24 hours. The amount of data accumulating in the SRL over this period would be:

(Application peak write rate x Duration of peak) +
(Application off-peak write rate x Duration of off peak).

In this case, the calculation would appear as follows:

```
1 GB/hour x 12 hours+ 1/4 GB/hour x 12 = 15 GB
```

An adjustment of 25% is made to handle header information. Since the 24-hour downtime is already an extreme case, no additional adjustments are needed to handle other constraints. The result shows that the SRL should be at least 18.75 gigabytes.

Tuning Replication Performance 2

The important factors that affect VVR performance are the layout of the SRL and the sizing of the VVR buffers. This chapter explains how to decide on the lay out of the SRL, and size the VVR buffers. It also describes how to choose the value of other VVR tunables.

SRL Layout

This section explains how the SRL affects application performance and how a good SRL layout can improve performance.

How SRL Affects Performance

Incoming writes to a data volume on the Primary are written to the SRL, first, and then to the data volume. VVR manages writes in the same way, irrespective of the replication settings, including the mode of replication. Note that writes to different data volumes within the RVG are also written in the same SRL. Therefore, the SRL throughput may affect performance.

The following reasons could alleviate the degradation of performance:

- ✔ The writes to SRL are sequential, whereas, the writes to the data volumes are spatially random in most cases. Typically, sequential writes are processed faster than the random writes.
- ✔ The SRL is not used to process read operations performed by the application. If a large percentage of the operations are read operations, then the SRL is not busy at these times.

If the rate at which the application writes to the data volume is greater than the rate at which the SRL can process writes, then the application could become slow. The following sections explain how to lay out the SRL to improve performance.

Striping the SRL

Striping the SRL over several physical disks to increase the available bandwidth can improve performance.

Choosing Disks for the SRL

It is recommended that there be no overlap between the physical disks comprising the SRL and those comprising the data volumes, because all write requests to VVR result in a write to both the SRL and the requested data volume. Any such overlap is guaranteed to lead to major performance problems, as the disk head thrashes between the SRL and data sections of the disk. Slowdowns of over 100% can be expected.

Mirroring the SRL

It is recommended that the SRL be mirrored to improve its reliability. The loss of the SRL immediately stops replication. The only way to recover from this is to perform a full resynchronization, which is a time-consuming procedure to be avoided whenever possible. Under certain circumstances, the loss of the SRL may even cause loss of the data volumes. The risk of this failure can be minimized by mirroring the SRL.

Tuning VVR

This section describes how to adjust the tunable parameters that control the system resources used by VVR. Depending on the system resources that are available, adjustments may be required to the values of some tunable parameters to optimize performance.

For instructions on changing the value of tunables, refer to the V*ERITAS Volume Replicator Administrator's Guide*.

VVR Buffer Space

VVR uses the following buffers to replicate data:

- ✔ Write Buffer Space on the Primary
- ✔ Readback Buffer Space on the Primary
- ✔ Buffer Space on the Secondary

Write Buffer Space on the Primary

When a write is issued, a write buffer is allocated from the write buffer space on the Primary. The buffer is not released until the data has been written to the Primary SRL and sent to all the Secondaries in synchronous mode. If the Secondaries in asynchronous mode cannot keep up with the application write rate, the data to be sent to the Secondary starts accumulating in the write-buffer space on the Primary. As a result, write-buffer space on the Primary becomes low. Then, VVR begins to free some buffers and postpones sending the data to the Secondaries in asynchronous mode. As a result, more space is freed up for incoming write requests so that they are not delayed.

Readback Buffer Space on the Primary

When VVR is ready to send the freed request to the Secondary, the freed requests are read back from the SRL. The data from the SRL is read back in to the Readback buffer space on the Primary.

As discussed in "Choosing the Mode of Replication" on page 6, the need to read back data from the SRL has an impact on write latency because more non-sequential I/O is performed on the SRL. Reading back data from the SRL also increases the load on the system and slows the rate at which data is sent to the Secondaries.

Note that the write buffer is freed only if the mode of replication is asynchronous; the writes do not have to be read back from the SRL when replicating in synchronous mode.

Buffer Space on the Secondary

The writes from the Primary are received in to the buffer space on the Secondary. The write is then written to the Secondary data volume from this buffer space. A write on the Primary can complete before the write to the Secondary data volume completes, even in synchronous mode of replication. However, if the Secondary is low on buffer space, it rejects new writes from the Primary thereby slowing down the Primary. On the Primary this appears as an inability to send requests over the network. The results are identical to those pertaining to insufficient network bandwidth.

For Secondaries in asynchronous mode, there may be no limit to how far Secondary data volumes can fall behind unless the mechanisms discussed in "Choosing Latency and SRL Protection" on page 11 are in force. Hence, if all the Secondaries are replicating in asynchronous mode, the application may not slow down; if there are Secondaries in synchronous mode, the write rate of the application reduces.

Tunable Parameters for the VVR Buffer Spaces

The amount of buffer space available to VVR affects the application and replication performance. You can use the following tunables to manage buffer space according to your requirements:

- `vol_rvio_maxpool_sz`
- `vol_min_lowmem_sz`
- `vol_max_rdback_sz`
- `vol_max_nmpool_sz`

Use the `vxmemstat` command to monitor the buffer space used by VVR. The following sections describe each of the above tunables. For instructions on changing the value of the tunable parameters, refer to the *VERITAS Volume Replicator Administrator's Guide*.

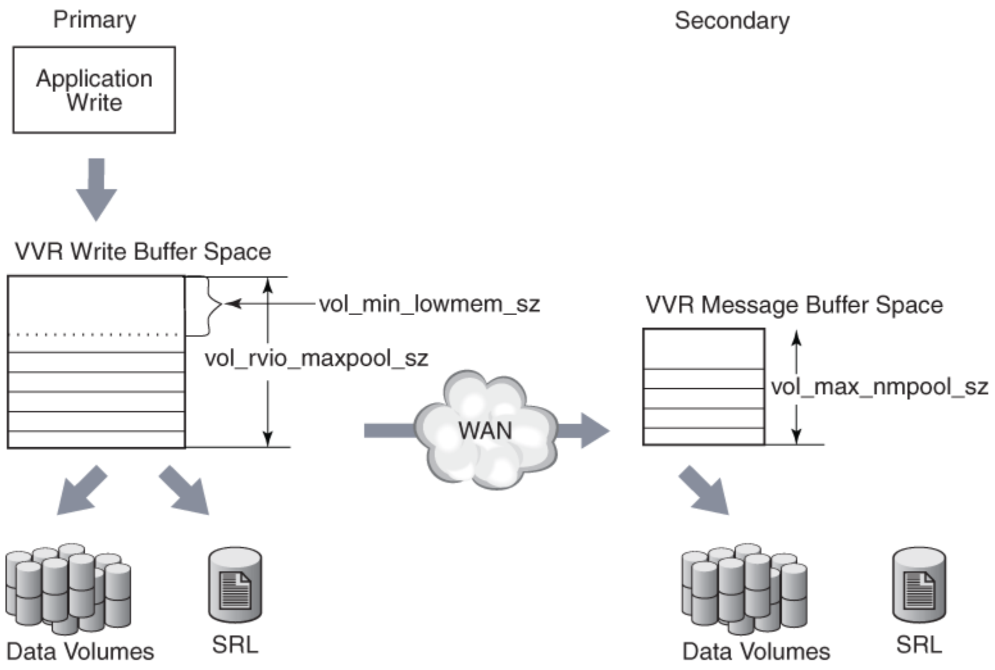
Tunable Parameters for the Write Buffer Space on the Primary

The following tunable parameters control the write buffer space on the Primary:

- `vol_rvio_maxpool_sz`
- `vol_min_lowmem_sz`

The amount of buffer space that can be allocated within the operating system to handle incoming writes is defined by the tunable `vol_rvio_maxpool_sz`.

How VVR Uses Buffers Between the Primary and Secondary



If the available buffer space is not sufficient to process the write request, writes are held up. VVR must wait for current writes to complete and release the memory being used before processing new writes. Furthermore, when the buffer space is low, VVR frees buffers early, requiring VVR to read back the write from the SRL, as explained in "Write Buffer Space on the Primary" on page 27. Both these problems can be alleviated by increasing `vol_rvio_maxpool_sz`. By setting the `vol_rvio_maxpool_sz` to be large enough to hold the incoming writes, you can increase the number of concurrent writes and reduce the number of readbacks from the SRL. When decreasing the value of the `vol_rvio_maxpool_sz` tunable, stop all the RVGs on the system on which you are performing this operation.

When deciding whether or not a given write is freed early and read back later, VVR looks at the amount of buffer space available, and frees the write if the amount is below a threshold, which is about 520 kilobytes, twice the size of a maximum write. This threshold is too low in some cases, which results in buffers being held for a long time. New writes cannot be performed because of lack of buffer space.

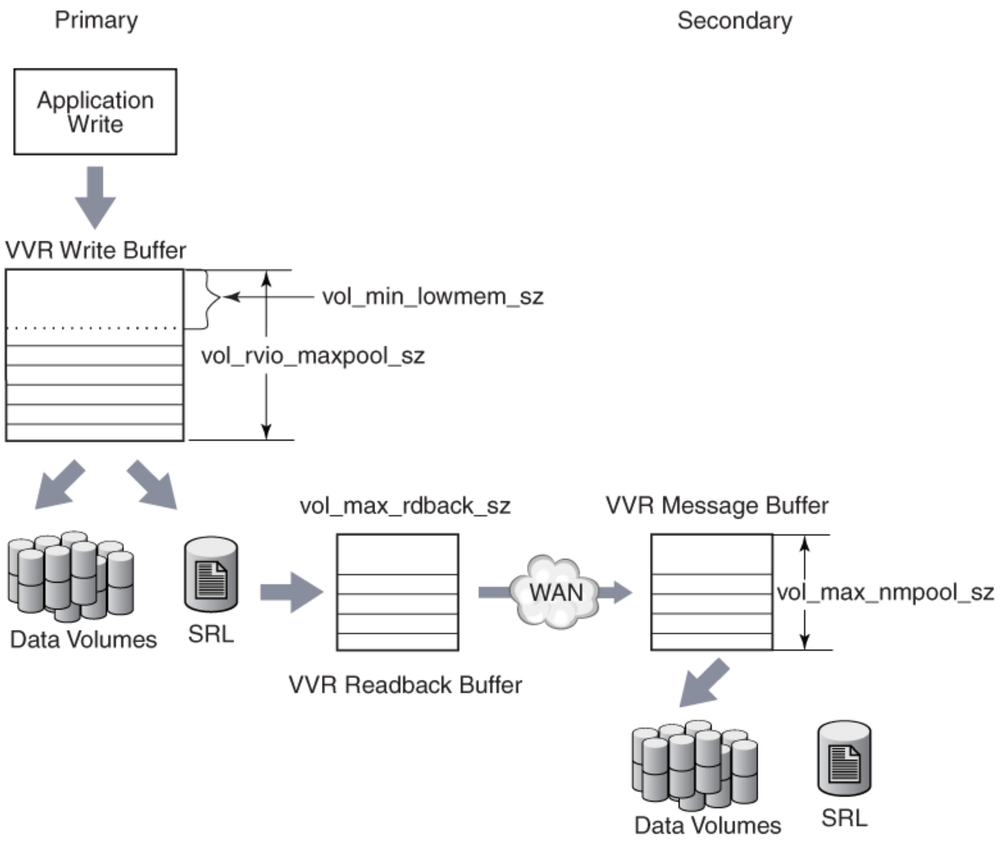
You can raise the threshold by increasing the value of the tunable `vol_min_lowmem_sz`. It should be set to, at least, 3 x N x I, but not less than 520K, where N is the number of concurrent writes to replicated volumes, and I is the average I/O size, rounded up to 8 kilobytes. The `vol_min_lowmem_sz` tunable is auto-tunable and depending on the incoming writes, VVR will increase or decrease the tunable value. The value that you specify for the tunable, using the `vxtune` utility or the system-specific interface, will be used as an initial value and could change depending on the application write behavior.

Note that this tunable is used only when replicating in asynchronous mode because SRL is not read back when replicating in synchronous mode.

Use the `vxrvg stats` command to determine the maximum concurrency (N) and average write size (I).

Tunable Parameter for the Readback Buffer Space

The amount of buffer space available for these readbacks is defined by the tunable, `vol_max_rdback_sz`. To accommodate reading back more data, increase the value of `vol_max_rdback_sz`. You may need to increase this value if you have multiple Secondaries in asynchronous mode for one or more RVGs.



Use the `vxmemstat` command to monitor the buffer space. If the output indicates that the amount of space available is completely used, increase the value of the `vol_max_rdback_sz` tunable to improve readback performance. When decreasing the value of the `vol_max_rdback_sz` tunable, pause replication to all the Secondaries to which VVR is currently replicating.

Tunable Parameters for the Buffer Space on the Secondary

The amount of buffer space available for requests coming in to the Secondary over the network is determined by the VVR tunable, `vol_max_nmpool_sz`. VVR allocates separate buffer space for each Secondary RVG, the size of which is equal to the value of the tunable `vol_max_nmpool_sz`. The buffer space on the Secondary must be large enough to prevent slowing the network transfers excessively.

If the buffer is too large, it can cause problems. When a write arrives at the Secondary, the Secondary sends an acknowledgement to the Primary so that the Primary knows the transfer is complete. When the write is written to the data volume on the Secondary, the Secondary sends another acknowledgement, which tells the Primary that the write can be discarded from the SRL. However, if this second acknowledgement is not sent within one minute, the Primary disconnects the RLINK. The RLINK reconnects immediately but this causes disruption of the network flow and potentially other problems. Thus, the buffer space on the Secondary should be sized in such a way that no write can remain in it for one minute. This size depends on the rate at which the data can be written to the disks, which is dependent on the disks themselves, the I/O buses, the load on the system, and the nature of the writes (random or sequential, small or large).

If the write rate is W megabytes/second, the size of the buffer should be no greater than W * 50 megabytes, that is, 50 seconds' worth of writes.

There are various ways to measure W. If the disks and volume layouts on the Secondary are comparable to those on the Primary and you have I/O statistics from the Primary before replication was implemented, these statistics can serve to arrive at the maximum write rate.

Alternatively, if replication has already been implemented, start by sizing the buffer space on the Secondary to be large enough to avoid timeout and memory errors.

While replication is active at the peak rate, run the following command and make sure there are no memory errors and the number of timeout errors is small:

```
# vxrlink -g diskgroup -i5 stats rlink_name
```

Then, run the `vxstat` command to get the lowest write rate:

```
# vxstat -g diskgroup -i5
```

The output looks similar to this:

```
                      OPERATIONS          BLOCKS       AVG TIME(ms)
TYP NAME            READ      WRITE       READ      WRITE    READ  WRITE

Mon 29 Sep 2003 07:33:07 AM PDT
vol archive            0        750          0        750     0.0    9.0
vol archive-L01        0        384          0        384     0.0    5.9
vol archive-L02        0        366          0        366     0.0   12.1
vol ora02              0        450          0        900     0.0   11.1
vol ora03              0          0          0          0     0.0    0.0
vol ora04              0          0          0          0     0.0    0.0


Mon 29 Sep 2003 07:33:12 AM PDT
vol archive            0        495          0        495     0.0   10.1
vol archive-L01        0        256          0        256     0.0    5.9
vol archive-L02        0        239          0        239     0.0   14.4
vol ora02              0        494          0        988     0.0   10.0
vol ora03              0          0          0          0     0.0    0.0
vol ora04              0          0          0          0     0.0    0.0
```

For each interval, add the numbers in the blocks written column, but omit any subvolumes. For example, `archive-L01`, and `archive-L02` are subvolumes of the volume `archive`. The statistics of the writes to the subvolumes are included in the statistics for the volume `archive`. You may vary the interval, the total time you run the test, and the number of times you run the test according to your needs. In this example, the interval is 5 seconds and the count is in blocks, hence on a machine with 2 kilobytes of block size, the number of megabytes per interval, M, is (total * 2048)/(1024*1024), where total is the sum for one interval. Hence, for one second the number of megabytes is M/5 and the size of the buffer is (M/5)*50. If there is more than one Primary, do not increase the buffer size beyond this number.

DCM Replay Block Size

When the Data Change Map (DCM) is being replayed, data is sent to the Secondary in blocks. The tunable `vol_dcm_replay_size` enables you to configure the size of the DCM replay blocks according to your network conditions. The default value of `vol_dcm_replay_size` is 256K. Decreasing the value of the tunable `vol_dcm_replay_size` may improve performance in a high latency environment.

Heartbeat Timeout

VVR uses a heartbeat mechanism to detect communication failures between the Primary and the Secondary hosts. The RLINKs connect after the heartbeats are exchanged between the Primary and the Secondary. The RLINK remains connected while the heartbeats continue to be acknowledged by the remote host. The maximum interval during which heartbeats can remain unacknowledged is known as the heartbeat timeout value. If the heartbeat is not acknowledged within the specified timeout value, VVR disconnects the RLINK.

The tunable `vol_nm_hb_timeout` enables you to specify the heartbeat timeout value. The default is 10 seconds. For a high latency network, increasing the default value of the tunable `vol_nm_hb_timeout` prevents the RLINKs from experiencing false disconnects.

Memory Chunk Size

The tunable `voliomem_chunk_size` enables you to configure the granularity of memory chunks used by VVR when allocating or releasing system memory. A memory chunk is a contiguous piece of memory that can be written to the disk in one operation. If the write size of the application is larger than the memory chunk size then the write is split resulting in multiple operations, which can reduce performance.

The default memory chunk size is 64K. For applications performing large writes, increase the size of `voliomem_chunk_size` to improve replication performance. The maximum value of `voliomem_chunk_size` is 128K.

VVR and Network Address Translation Firewall

VVR uses a heartbeat mechanism to detect communication failures between the Primary and the Secondary hosts. VVR uses IP addresses in the heartbeat message to send heartbeat acknowledgement.

When replicating over a Network Address Translation (NAT) based firewall, VVR must use the translated IP address, instead of the IP address in the heartbeat message. If the IP address in the heartbeat message is used, the heartbeat acknowledgement is dropped at the firewall and replication does not start.

The tunable `vol_vvr_use_nat` enables VVR to use the translated address from the received message so that VVR can communicate over a NAT-based firewall. Set this tunable to `1` only if there is a NAT-based firewall in the configuration.

Glossary

asynchronous

Asynchronous mode queues writes persistently and holds them at the Primary for later transmission. This mode can deal with temporary outages of the network or the Secondary node without impacting the application. Asynchronous mode is useful when it is acceptable for the Secondary not to be up to date.

Automatic Synchronization

A feature of VVR that automatically synchronizes the Secondary without interrupting the Primary.

buffer space

The memory used by VVR to process writes and perform replication.

checkpoint

A feature of VVR that synchronizes the Secondary using a block-level backup and restore method without interrupting the Primary.

checkpoint end

Denotes the end point of the data for a Primary checkpoint. After a checkpoint attach of the RLINK, the Secondary becomes consistent when the checkpoint end is reached.

checkpoint start

Denotes the starting point of the data for a Primary checkpoint. After the checkpoint attach of the RLINK, VVR starts sending writes performed after the checkpoint start.

consistent

A flag indicating that data is recoverable by the system or application using it. For example, if the data contains a file system, the data is consistent if `fsck` can be run successfully on it. If the data contains a database, the data is consistent if the database recovery program can be run successfully and the database restarted.

Data Change Map (DCM)

An object containing a bitmap that can be optionally associated with a data volume on the Primary RVG. The bits represent regions of data that are different between the Primary and the Secondary.

data volume

A volume that is associated with an RVG and contains application data.

heartbeat protocol

The heartbeat protocol ensures that the nodes in an RDS will detect any network outage or a node crash. Once the Primary detects an outage, it begins a periodic attempt to reestablish a connection which continues until successful. Upon success, replication resumes automatically unless some interim administrative command or error prevents it.

inconsistent flag

A flag used for the following conditions:

Inconsistent data on the Secondary, which indicates that the Secondary is not a viable takeover candidate.

An RLINK in the FAIL state. Either the Secondary must be restored from a checkpoint or a full resynchronization must be performed.

This flag is set during the DCM replay or a Primary checkpoint; it is cleared automatically when the DCM replay completes or the checkpoint end is reached.

latency high mark

Specifies the maximum number of waiting updates in the SRL before latency protection becomes active and writes stall or fail depending on the mode of the latency protection.

latency low mark

Specifies the number of writes in the SRL when latency protection becomes inactive and writes proceed without waiting.

latency protection

For RLINKs operating in asynchronous mode, which may fall behind, the latency protection attribute (`latencyprot`) of the RLINK is used to limit the maximum number of outstanding write requests. If latency protection is enabled, the maximum number of outstanding write requests cannot exceed the value set in `latency_high_mark`.

latencyprot

See latency protection.

modes of replication

VVR replicates in *asynchronous* and *synchronous* modes. Each mode uses a different process to replicate data, and each deals differently with network conditions. See asynchronous and synchronous for more details.

Primary checkpoint

A method for synchronizing a Secondary with the Primary without stopping the application. A command marks the start of the checkpoint. All Primary data volumes are backed-up and then the end of the checkpoint is marked. The data is restored on the Secondary and the Primary can then begin from the start of the checkpoint. The Secondary becomes consistent when the end of the checkpoint is reached.

Primary node

The node on which the Primary RVG resides.

Primary node SRL overflow

Because the Primary SRL is finite, prolonged halts in update activity to any RLINK can exceed the SRL's ability to maintain all the necessary update history to bring an RLINK up to date. When this occurs, the RLINK is marked as STALE and requires manual recovery before replication can proceed.

Primary Replicated Volume Group

In the VVR replication scheme, one copy of each replicated volume is designated the Primary volume, and the other copies are called Secondary volumes. Secondary volumes reside on different network hosts. Each Secondary volume group is represented on the Primary node by a single RLINK object, which serves as a proxy on the Primary node for the Secondary volume. Thus, when the Primary RVG needs to write data to a Secondary volume, it performs a write to the corresponding RLINK object. That object is then responsible for sending the request to the Secondary node. The RLINK object also provides an interface that allows for the sending of control messages to the remote RVG.

RDS

See Replicated Data Set

readback

Volume Replicator allocates memory for application writes from `voliomem pool` and frees the memory when the write request is written to the Primary and all Secondary data volumes. When the memory pool space becomes low, VVR frees some memory space for new write requests by postponing sending the data across asynchronous RLINKs. Later, these write requests are read back from the SRL when the RLINK is ready to send them. Such reads are called readbacks.

Replicated Data Set

The group of the RVG on a Primary and its corresponding Secondary hosts.

Replicated Volume Group

A component of VVR that is made up of a set of data volumes, one or more RLINKs, and an SRL.

Resynchronization

The process of making the data on the Secondary identical to the data on the Primary. Resynchronization can be done without stopping the application by using a primary checkpoint or by using the Auto Synchronization feature.

RLINK

An RLINK represents the communication link to the counterpart of the RVG on another node. At the Primary node a replicated volume object has one RLINK for each of its network mirrors. On the Secondary node a Replicated Volume Group has a single RLINK object that links it to its Primary.

RVG

See Replicated Volume Group

Secondary checkpoint

A method for backing up a Secondary to allow for quick recovery in the event of a data volume failure on the Secondary.

Allows volume-level backups of the RVG to be taken at the Secondary node.

Secondary node

The node to which the Primary data volumes are replicating.

Secondary Replicated Volume Group

See Replicated Volume Group.

Secondary SRL volume failure

Secondary SRL volume failure is not a serious error because all the data is still available on the Primary. The failure will, however, cause the RLINK to be placed in a PAUSED state, just as if a system administrator had paused the volume.

SRL

See Storage Replicator Log.

SRL overflow protection

For RLINKs with latency protection disabled, a final degree of protection is provided by SRL overflow protection (`srlprot`). When SRL overflow protection is enabled, the RLINK is protected from having so many outstanding write requests that it would overflow its SRL and require a full resynchronization.

STALE

If an RLINK is STALE, the corresponding Secondary requires a full resynchronization to make it usable. RLINKS can enter the STALE state when they are detached manually by issuing the `vxrlink det` command or as a result of a Primary failure. In addition, when an RLINK is associated for the first time, its state changes from UNASSOC to STALE.

Storage Replicator Log (SRL)

Writes to the Primary RVG are saved in an SRL on the Primary side. The SRL is used to aid in error recovery, as well as to buffer writes when the system operates in asynchronous mode. Each write to a data volume in the RVG generates two write requests: one to the Secondary SRL and another to the Primary SRL.

synchronous

In synchronous mode, the Secondary is kept up to date with the Primary by waiting for each write request to reach the Secondary before the application sees the successful completion of the write on the Primary.

throttling

A mechanism that delays incoming writes.

update

Data from the Primary corresponding to the application writes sent to the Secondary for replicating the writes is referred to as an update.

Volume Replicator Objects

The objects used for replication such as Replicated Volume Group, Storage Replicator Log (SRL), RLINK, and Data Change Map (DCM).

Index